



به نام خدا



## فاز دوم پروژه کامپایلرها و زبان های برنامه‌نویسی

پاییز ۹۸

مهلت تحویل: ۲۷ آبان

در فاز قبل، تحلیل‌گر لغوی و نحوی را برای زبان `acton` پیاده‌سازی کردید. در این فاز شما باید اطلاعات مربوط به `actor` ها، `msghandler` های هر یک از آن‌ها، `scope` های برنامه و متغیرهای آن‌ها را جمع‌آوری کرده و در جدول علائم ذخیره کنید. برای این کار شما باید درخت `AST` مربوط به برنامه را در این فاز تشکیل دهید. به منظور سادگی کار شما، یک شمای کلی از جدول علائم در اختیارتان قرار گرفته که بایستی آن را به نحو مناسب استفاده و پر نمایید. همچنین نودهای درخت `AST` در اختیار شما قرار گرفته که باید با توجه به آن‌ها با تعریف `action` های مناسب در گرامرتان، درخت را بسازید.

در این فاز باید موارد زیر را بررسی کرده و در صورت خطا، پیام مناسب را به فرمتی که برای هر خطا گفته شده به ترتیب شماره‌ی خط پرینت کنید و به بررسی کد تا انتهای فایل ادامه دهید. فرمت کلی هر یک از خطاهایی که باید نمایش دهید به شکل زیر است:

Line:<LineNumber>:<ErrorMessage>

مثال:

Line:21:Redifinition of actor A

که `ErrorMessage` برای هر یک از موارد زیر تعریف می‌شود.

Symbol Table

Abstract Syntax Tree

## ۱. عدم وجود دو actor با نام یکسان در برنامه

در صورت خطا در این قسمت باید خطای مربوطه اعلام گردد و یک نام موقت برای اکتور کنونی در نظر گرفته شود. برای اسم دادن دوباره به اکتور، از الگویی استفاده کنید که برخوردی با نام اکتور از پیش موجود در برنامه نداشته باشد.

ErrorMessage: Redefinition of actor <ActorName>

## ۲. عدم وجود دو متغیر با نام یکسان در یک scope

در صورت خطا همانند بخش قبل عمل کنید. دقت شود که با توجه به اینکه actorvar overloading نداریم، این خطا شامل actorvar ها و knownactor های اکتوری که از اکتور دیگر ارث‌بری می‌کند نیز می‌شود.

ErrorMessage: Redefinition of variable <VariableName>

## ۳. عدم تعریف دو msghandler هم‌نام در یک actor

در صورت خطا همانند بخش قبل عمل کنید. دقت شود که با توجه به اینکه msghandler overriding نداریم، این خطا شامل msghandler هایی که یک اکتور از آن‌ها ارث‌بری می‌کند نیز می‌شود. initial از این مورد مستثنی است.

ErrorMessage: Redefinition of msghandler <MsgHandlerName>

## ۴. عدم تعریف آرایه با طول مساوی صفر

ErrorMessage: Array size must be positive

## ۵. عدم تعریف اکتور با طول صف مساوی صفر

ErrorMessage: Queue size must be positive

## ۶. وراثت حلقوی

ErrorItemMessage: Cyclic inheritance involving actor <ActorName>

توجه کنید که این خطا تنها باید از اولین اکتور داخل یک حلقه ی وراثت گرفته شود.

پس از پیمایش کامل کد در حالتی که ورودی هیچ خطایی نداشته باشد، شما باید پیمایش preorder درخت AST را در خروجی چاپ کنید. (به ازای هر نود، خروجی تابع toString آن را در یک خط خروجی چاپ کنید.)

### نکات مهم:

- گرامر زبان در اختیارتان قرار داده شده است. می توانید از آن یا گرامری که خودتان نوشتید استفاده کنید. استفاده از گرامر خودتان اندکی نمره امتیازی خواهد داشت اما توصیه می شود در صورت عدم اطمینان از آن از گرامری که در اختیارتان قرار می گیرد استفاده کنید. اگر از گرامر خودتان استفاده می کنید، قاعده ی شروع گرامر باید program باشد.
- با توجه به اینکه خروجی های شما به صورت خودکار تست می شوند، لازم است تا به این نکات در هنگام آپلود توجه کنید: یک فایل با فرمت stdID1\_stdID2.zip (اگر گروهی تک نفره هستید stdID.zip) آپلود کنید که در آن فایل های شما قرار دارد. فولدر main که شامل کدهای AST، symbol table، visitor و فایل اصلی Acton.java میباشد در اختیار شما قرار داده شده است و باید مستقیماً در سورس پروژه قرار گیرد.
- در هنگام اجرای تست کیس ها، فایل Acton.java اجرا خواهد شد. در این فایل پس از فراخوانی قانون اول گرامر (که پارسر زبان Acton را اجرا خواهد کرد و درخت ast در حین اجرای آن ساخته خواهد شد)، باید کد مربوط به پیمایش ast را قرار دهید. در نهایت کافیسیت علاوه بر کامپایل کردن فایل های جاوا و گرامرتان و این فایل همانند قبل، این فایل را با یک آرگومان (اسم فایل تستتان، مثلاً test.act) اجرا کنید.